REMARKS

Claims 1, 2, 5, 7, 8, 11, and 13-23 are currently pending. Claims 1, 2, 5, 7, 8, and 11 have been amended. Claims 16-23 have been added. The support for the amendment of claim 1 is found in original claims 1, 3, 4, and 6, page 5, line 20, through page 6, line 2, of Applicant's specification, and Figure 2. The support for the amendment of claim 7 is found in original claims 9, 10, and 12, page 5, line 20, through page 6, line 2, of Applicant's specification, and Figure 2. The support for new claims 16-23 is found in page 5, line 20, through page 6, line 2, of Applicant's specification and original claims 1 and 6. It is respectfully submitted that no new matter has been added.

Applicant has disclosed the following in the background of the invention:

In the field of this invention it is known that `Java 2` includes a significantly enhanced security model, compared to previous Java Virtual Machines (JVMs). This new model can restrict the behaviour of a Java applet or application to a clearly defined set of safe actions. This allows a user to download application code from the internet or across a computer network, and run the code in a previously installed JVM. The user can be confident that the application will be assigned the required privileges to function correctly, but to neither damage the user's machine nor divulge sensitive information held on that machine to others via the network or internet. **However, a problem with this approach is that the JVM itself must retain its security integrity in order to ensure downloaded code is restricted in this way. If a malicious user (hacker) has been able to gain access to the user's machine outside of the JVM environment and alter the behaviour of the JVM the whole Java security model is undermined.** For example, the hacker could alter the privileges assigned for software code from a specific source, thereby allowing subsequently downloaded code from this source to function beyond the limits otherwise set by the JVM, and such enhanced privileges could easily be configured to compromise the security integrity of the user's machine. Similarly, the hacker could disable the security code altogether, or worst still insert destructive routines into the core of the JVM which could be activated by an external trigger, such as specific time/date, or when other (possibly harmless) code is being executed. It is clear that with this malicious activity, early detection of such a compromise of the JVM core would be very useful, and could prevent more serious subsequent damage. If a malicious user decides to attack a machine, the JVM is an obvious target due to its significance in relation to web-based applications, servers and the like. Therefore the security integrity of the JVM is a highly significant factor in the security of the computer as a whole. **A need therefore exists for a software verification system, method and computer program element wherein the abovementioned disadvantages may be alleviated.**

Above, Applicant has defined a problem that McManis does not appear to have recognized. Further, Applicant has provided a solution to Applicant's identified problem.

It appears that in McManis, the verifier uses the public key provided by calling procedure A to decode the digital signature in procedure B (see Fig. 3A) and the verifier uses the public key provided by calling procedure B to decode the digital signature in procedure A (see Fig. 3B) and it appears that McManis provides no disclosure or fair suggestion of obtaining a public key from a virtual machine loader.

The Patent Office rejected claims 1, 2, 5, 7, 8, 11, and 13-15 under 35 U.S.C. 103(a) as being unpatentable over McManis, "System and Method for Protecting Use of Dynamically Linked Executable Modules," U.S. Patent No. 5,757,914.

Claim 1 recites "A verification system for a computer software installation, comprising a primary library file, the primary library file having a digital signature; a loader program that obtains a digital signature key and further loads the primary library file, wherein, if a public key cannot be obtained via a virtual machine provider, the digital signature key is a hidden public key internal to the loader program and, if a public key can be obtained via the virtual machine provider, the digital signature key is the public key obtained via the virtual machine provider; and a plurality of secondary files referenced by the primary library file, each of the plurality of secondary files having a digital signature; wherein the loader program verifies and selectively loads the primary library file by comparing the obtained digital signature key with the digital signature of the primary library file, the primary library file subsequently verifying and selectively loading the plurality of secondary files by calling the loader program to compare the obtained digital signature key with the digital signature of each of the plurality of secondary files, wherein the computer software installation is a virtual machine installation."

Claim 7 recites "A verification method for a computer software installation, the method comprising the steps of launching a loader program arranged to load files; if a public key is available from an internet site of a virtual machine provider, using the public key as a digital signature key; if a public is not available from the internet site of the virtual machine provider, using a hidden key as the digital signature key; using the loader program to verify the authenticity of a digital signature incorporated in a primary library file by comparing said digital signature with the digital signature key; selectively loading the primary library file in dependence upon the successful verification of its digital signature; using the primary library file and the loader program to verify the authenticity of digital signatures incorporated in each of a plurality of secondary files by comparing them with the digital signature key; and, selectively loading the plurality of secondary files in dependence upon the successful verification of their digital signatures, wherein the computer software installation is a virtual machine installation."

Applicant has identified problems with the current art; e.g., a hacker can alter the behavior of the JVM outside the JVM environment so as to undermine the whole Java security model (page 1, line 25, through page 2, line 4) such as by disabling the security code or by inserting destructive routines into the core of the JVM (page 2, lines 13-17). The present invention provides a scheme for verification of the authenticity of a JVM using digital signatures and offers advantages. These advantages include 1) enhanced security of the JVM, 2) greater user confidence in the correct function of Java applications, and 3) improved detection of incorrect or damaged JVM installations (page 9, line 20, through page 10, line 2, of Applicant's specification).

The Patent Office asserted (Page 3, line 8, through page 4, line 3, of the Office Action mailed March 07, 2006) "McManis discloses: *a primary library file having a digital signature* (McManis, col. 1, line 65 – col. 2, line 11; col. 1, lines 48-63); *a loader program arranged to obtain a digital signature key and further arranged to load the primary library file, wherein, if a public key internal to the loader program and, if a public key can be obtained via the virtual* *machine provider, the digital signature key is the public key obtained via the virtual machine provider* (McManis, fig. 1, elems. 110, 112; col. 2, lines 22-37, 40-43). The verifier is a "loader program" as it enables the loading of each program module, including the primary program module. Furthermore, the loader program possesses the "hidden" public key (note: McManis never discloses an open display of the key), enabling the loader program to perform the processing of the digital signatures. McManis does not disclose that the key could be obtained from a virtual machine provider. The public key of the loader program is used even if it couldn't be obtained by from a virtual machine provider. *and a plurality of secondary files arranged to be referenced by the primary library file, each of the plurality of secondary files having a digital signature* (McManis, fig. 1, elems. 116, 118, 120; col. 2, lines 1, 2; col. 3, lines 17-21)."

The claimed invention recites "wherein the loader program is arranged to verify and selectively load the primary library file by comparing the obtained digital signature key with the digital signature of the primary library file" and "selectively loading the plurality of secondary files in dependence upon the successful verification of their digital signatures," whereas McManis is limited to the mutual verification by two applications. Despite the Patent Office's assertion, McManis's verifier is not a "loader program." Just because a software module is able to verify the authenticity of an application does not mean that it will also load such application.

A loader, as defined by http://www.webopedia.com/TERM/l/loader.html, is "an operating system utility that copies programs from a storage device to main memory, where they can be executed. In addition to copying a program into main memory, the loader can also replace virtual addresses with physical addresses. Most loaders are transparent, i.e., you cannot directly execute them, but the operating system uses them when necessary."

McManis discloses (abstract) "The program module verifier responds to procedure calls by verifying the authenticity of any specified program module and by returning a verification confirmation or denial. When the program module verifier fails to verify the authenticity of a program module, the calling program module throws an exception and aborts its execution."

A loader is not a verifier. McManis does not disclose or fairly suggest a loader.

The Patent Office (page 4, lines 4-22, of the Office Action mailed March 07, 2006) asserts "McManis shows the operation of his system in a slice of time, wherein every file is verified (McManis, col. 1, line 65, - col. 2, line 4; col. 2, lines 53-55). He does not disclose, as can be suggested by the claims, that the primary file is initially verified before being loaded and then secondary files are verified and loaded. However, McManis discloses that every file, including the primary file, contains a digital signature and that the digital signature is necessary to verify the authenticity of every called file before that file is loaded (McManis, col. 1, line 65 – col. 2, line 44). It would have been obvious to one of ordinary skill in the art to arrange, at the time of the initialization loading of the primary file, for the loader program to load the primary file by verifying the digital signature of the primary file. This would have been obvious because one of ordinary skill in the art would have been motivated to verify the authenticity of the primary file, as taught by McManis, when the primary file is initially loaded – so as to protect the system's integrity at all times. Thus the qualification of McManis discloses:

Wherein the loader program is arranged to verify and selectively load the primary library file by comparing the obtained digital signature key with the digital signature of the primary library file, the primary library file being further arranged to subsequently verify and selectively load the plurality of secondary files by calling the loader program to compare the obtained digital signature of each of the plurality of secondary files (McManis, col. 1, line 65 – col. 2, line 44; fig. 1, 2).

McManis does not disclose the details regarding the initialization of his system, but instead shows how his loader program enables the loading of the primary and secondary files via the verification of digital signatures. Consequently, McManis does not disclose an initial digital signature verification of the primary file by the loader program, an initial loading of the primary file by the loader program".

Applicant asserts that McManis discloses a verifier that is usable by a calling application and a called application, but does not disclose a loader arranged to obtain a digital signature key and further arranged to load the primary library file, as claimed. Since McManis is concerned with preventing use or export of certain cryptographic routines, trade secret functions, and functions protected by contract (column 1, lines 37-63), McManis is not concerned with the basic software but with certain called routines and so it not concerned with the initial loading of a base application, such as a JVM.

The Patent Office (page 5, lines 1-11, of the Office Action mailed March 07, 2006) further asserts "Regarding claim 1, the qualification of McManis does not disclose that the *software installation is a virtual machine installation.* However, the system of McManis, assigned to Sun Microsystems, Inc., is disclosed as being operable on "virtually any type of computer," including architecturally distinct systems such as Sun workstations, IBM compatible computers, and Macintosh computers. It would have been obvious to one of ordinary skill in the art, based upon logical reasoning, to employ a virtual machine installation in the system of McManis. This would have been obvious because one of ordinary skill in the art would have logically recognized that a virtual machine installation would allow the system of McManis to be employed on such as diverse and distinct set of architectures. Thus the qualification of McManis discloses: *wherein the software installation is a virtual machine installation.*

Applicant, in response to the above paragraph, again asserts that McManis does not disclose loading and does not disclose loading as being part of the verification process. McManis does not disclose that the primary file is verified by a loading program. McManis discloses a mutual verification process where a calling application verifies the digital signature of a called application and the called application verifies the digital signature of the calling application (e.g., Figure 2). The first problem McManis addresses concerns the isolation of cryptographic routines to prevent the export of sensitive technology (column 1, lines 37-57) so McManis would not be directed to the verification of a basic Java virtual machine, but at cryptographic routines that might be called by the JVM. The second problem McManis addresses concerns the situation where there is a desire to limit or prevent use of dynamically linkable modules so as to protect trade secrets or provide protection for contractual reasons so McManis would also not be directed to verification of a basic Java virtual machine.

McManis, contrary to the Patent Office's assertion (page 3, lines 10-11, of the Office Action mailed March 07, 2006), does not appear to disclose that every file has a digital signature. McManis appears only to disclose digital signatures for program modules (e.g., abstract, lines 1-4; column 1, line 66, through column 2, line 2; column 6, line 60, through column 7, line 36). Digital signatures do not appear to be an inherent feature of a file either. **Digital signature**, according to http://www.netlingo.com/lookup.cfm?term=digital%20signature , is defined as follows: "Like a written signature at the bottom of a page, this is a piece of code that **can be** attached to an e-mail message or an online transaction to prove that you are the person who sent the information. A digital signature is not to be confused with a sig file. It is an important component for e-commerce, since it provides authentication and an increased level of security. Thus, McManis does not disclose or fairly suggest that a program module verifier has a digital signature.

McManis is silent regarding a hidden public key and neither disclose or suggest the limitation "the loader program being arranged to use the hidden public key in the event that a public key cannot be obtained from the internet". Furthermore, McManis' discloses an embedded public key, but does not suggest or disclose both a hidden public key and a public key obtained via the internet. Thus, claims 1 and 7 are allowable over the prior art of record.

Although McManis discloses (abstract) each program module includes a digital signature and an executable procedure, McManis is silent regarding the program module verifier. It appears that McManis is unconcerned with the security issues that Applicant has recognized and provided a solution for regarding virtual machine installations. Applicant is concerned with providing security for the installation of a Java Virtual Machine (page 3, lines 4-6, and page 4, lines 22-27, of Applicant's specification) and notes that the prior art includes virtual machine enhanced security models that does not divulge sensitive information (page 1, lines 12-23, of Applicant's specification). Applicant asserts that McManis does not disclose or fairly suggest a Virtual Machine software installation, as found in claims 1 and 7.

Claim 2 recites "wherein after successful verification and selective loading of the at least one secondary file, the at least one secondary file is arranged to manage the verification and selective loading of the at least one tertiary file."

Claim 8 recites "after successful verification and selective loading of the at least one secondary file; using the at least one secondary file to manage the verification and selective loading of the at least one tertiary file."

In view of the above, allowance of claims 1, 2, 5, 7, 8, 11, and 13 is respectfully urged.

The Patent Office asserted (page 5, lines 13-20, of the Office Action mailed March 07, 2006) "Regarding claim 2, the modification of McManis discloses: *the plurality of files including at least one tertiary file referenced by at least one secondary file of the plurality secondary files, wherein after successful verification and selective loading of the at least one secondary file, the at least one secondary file is arranged to manage the verification and selective loading of the at least one tertiary file* (McManis, fig. 1, elems. 118, 120; col. 3, lines 12-21, 30-37). McManis discloses that each file can contain a plurality of procedure calls to other files, thus a secondary file may call a tertiary file."

Applicant asserts that McManis does not disclose or suggest the limitation "wherein after successful verification and selective loading of one of the at least one secondary file, the at least one secondary file is arranged to manage the verification and selective loading of the at least one tertiary file." Even if column 3, lines 12-21 and 30-37, figure 1, and elements 118 and 120 of McManis could be construed to as a suggestion for a tertiary file, McManis does not disclose or suggest that "at least one secondary file is arranged to manage the verification and selective loading of the at least one tertiary file." Thus, it is respectfully submitted that claims 2-5, 8 and 11 are allowable for this additional reason.

Claim 5 recites "comprising at least one administrator-configurable file" and "wherein the loader program is further arranged to verify the digital signature of the at least one administrator-configurable file using the private key."

Claim 11 recites "comprising at least one administrator-configurable file" and "wherein the loader program is further arranged to verify and selectively load the digital signature of the at least one administrator-configurable file using the private key."

The Patent Office asserted (page 6, lines 1-9 of the Office Action mailed March 07, 2006) "Regarding claim 5, the modification of McManis discloses that all files contain digital signatures so that they may be verified with a digital signature key (McManis, col. 2, lines 22-37). McManis further discloses that verifiable files may contain a number of portions, including a methods portion and a data portion. Each portion is verified by a separate digital signature (McManis, col. 4, lines 54-67). Thus, McManis discloses the digital signature key used to verify the file as being a combination of keys. These verifiable files are often authored, maintained, or updated ("administered") by others ("administrators"), which is why they are linked dynamically during program execution (McManis, col. 1, lines 10-27). Thus, the modification of McManis discloses at least one of the files as being an administrator configurable file."

Applicant asserts that McManis contains no teaching or suggestion of the limitation "wherein the loader program is further arranged to verify the digital signature of the at least one administrator-configurable file using the private key" and does not disclose an "administrator-configurable file." Thus, it is respectfully submitted that claims 5 and 11 are allowable over the prior art of record.

Specifically, as to claims 14 and 15, The Patent Office in the Final Office Action mailed October 21, 2005, (page 6, lines 12-13) asserted McManis "does not disclose that the public key is obtained from the internet."

Claim 22 recites "A system for a computer software installation, comprising: a virtual machine primary library file, the virtual machine primary library file having a digital signature; a loader program that obtains a digital signature key and further loads the virtual machine primary library file; and a plurality of secondary files referenced by the virtual machine primary library file, each of the plurality of secondary files having a digital signature; wherein **the loader program verifies and selectively loads the virtual machine primary library file by comparing the obtained digital signature key with the digital signature of the virtual machine primary library file**, the virtual machine primary library file subsequently verifying and selectively loading the plurality of secondary files by calling the loader program to compare the obtained digital signature key with the digital signature of each of the plurality of secondary files, wherein the computer software installation is a virtual machine installation."

The prior art of record does not teach or fairly disclose "**the loader program verifies and selectively loads the virtual machine primary library file by comparing the obtained digital signature key with the digital signature of the virtual machine primary library file.**" Thus, claims 22 and 23 are allowable over the prior art of record.

Regarding the Patent Office's Response to Arguments starting on page 7, line 7, of the Office Action mailed March 07, 2006.

(i) McManis does not disclose the verifier as a "loader program." The verifier of McManis does not carry out a loading function. Thus, the verifier is not a loader.

(ii) Applicant recognizes that security is providing for loading software modules in the current art, but has recognized a problem that current known methods for installing virtual machines are vulnerable (see Applicant's background of the invention). Applicant provides a solution to this identified problem.

The Patent Office's assertions regarding McManis relate roughly to the following disclosure from Applicant's background of the invention: "In the field of this invention it is known that `Java 2` includes a significantly enhanced security model, compared to previous Java

Virtual Machines (JVMs). This new model can restrict the behaviour of a Java applet or application to a clearly defined set of safe actions. This allows a user to download application code from the internet or across a computer network, and run the code in a previously installed JVM. The user can be confident that the application will be assigned the required privileges to function correctly, but to neither damage the user's machine nor divulge sensitive information held on that machine to others via the network or internet." McManis stops there.

Applicant continues (background of the invention): "**However, a problem with this approach is that the JVM itself must retain its security integrity in order to ensure downloaded code is restricted in this way. If a malicious user (hacker) has been able to gain access to the user's machine outside of the JVM environment and alter the behaviour of the JVM the whole Java security model is undermined.** For example, the hacker could alter the privileges assigned for software code from a specific source, thereby allowing subsequently downloaded code from this source to function beyond the limits otherwise set by the JVM, and such enhanced privileges could easily be configured to compromise the security integrity of the user's machine. Similarly, the hacker could disable the security code altogether, or worst still insert destructive routines into the core of the JVM which could be activated by an external trigger, such as specific time/date, or when other (possibly harmless) code is being executed. It is clear that with this malicious activity, early detection of such a compromise of the JVM core would be very useful, and could prevent more serious subsequent damage. If a malicious user decides to attack a machine, the JVM is an obvious target due to its significance in relation to web-based applications, servers and the like. Therefore the security integrity of the JVM is a highly significant factor in the security of the computer as a whole. **A need therefore exists for a software verification system, method and computer program element wherein the abovementioned disadvantages may be alleviated.**"

After identifying the problem, Applicant describes his solution in the detailed description of the invention.

Regarding the Patent Office's language on page 9, lines 1-12, of the Office Action dated March 07, 2006, Applicant directs the Patent Office's attention to new claims 18-21. Applicant is aware that "During patent examination, the pending claims must be "given their **broadest reasonable interpretation consistent with the specification.**" MPEP 2111.

Regarding the Patent Office assertion (page 10, lines 15-19, of the Office Action dated March 07, 2006) "Rather, the application has described numerous and very different computer elements ("operating system," "loader program," a "java.policy file") as able to "load" and for performing "loading." In light of such liberal and broad disclosure by the applicant respecting the terms "load" and "loading," as well as the claim language, the applicant's arguments are unpersuasive." Applicant notes that Figure 1 of Applicant's drawings illustrates a virtual machine installation methodology. A third-party application 10 or a Java launcher 20 loads the virtual machine dynamic link library 30. The virtual machine dynamic link library then loads second dynamic link libraries 40, 60 and secondary configuration files 50. "In a similar way the secondary DLL 60 may in turn load other DLLs and configuration files, such as the tertiary DLL 70. Each of the above files in the JVM installation has a digital signature attached to it" (page 5, line 25, through page 6, line 1, of the specification as filed). Configuration files 50 may include 'java.security' and 'java.policy' files. In Applicant's disclosure, a third-party application, Java launcher, or a dynamic link library may load a dynamic link library or a configuration file.

(iii) Applicant does disclose that in Applicant's invention, the launcher program loads and verifies the virtual machine DLL (page 7, lines 6-27). However, McManis, does not disclose loading. As discussed above, verifiers and loaders are not necessarily the same software entity. In the case of McManis, it appears that the verifier is separate from any loader. At least, McManis is silent regarding loaders.

(iv) McManis discloses (column 3, lines 12-21) "Each method or procedure 128 includes at least one verifier procedure call instruction 130 and instructions 132 for responding to a verification denial message received in response to the verifier procedure call, such as instructions for aborting execution of the procedure. The main application A procedure (128-A) in the first program module furthermore includes a procedure call 134 to an executable procedure (e.g., the main application B procedure 128-B) in the second procedure module." McManis discloses (column 3, lines 30-37) "More generally, in other embodiments of the invention the procedure call 130-B to the program module verifier is logically positioned in the second program module (and more generally, in all program modules that will be called by other program modules) prior to the completion point in each executable procedure in the second program module so as prevent completion of the execution of each such procedure if verification of the calling program is denied by the verifier." Applicant's claimed invention recites "wherein the loader program verifies and selectively loads the primary library file by comparing the obtained digital signature key with the digital signature of the primary library file, the primary library file subsequently verifying and selectively loading the plurality of secondary files by calling the loader program to compare the obtained digital signature key with the digital signature of each of the plurality of secondary files." In the cited passages of McManis, there does not appear to be disclosure of **a loader program, a primary library file, and a secondary library file**. Applicant requests that the Patent Office identify 1) **the loader program**, 2) **the primary library file,** ***and*** 3) **the secondary library file** in the cited passages of or elsewhere in McManis.

(v) The Patent Office did not answer the passage cited (page 12, lines 1-5, of the Office Action mailed March 07, 2006) from the preliminary amendment filed December 21, 2005. Quoting from page 12, lines 1-5, of the Office Action mailed March 07, 2006, "Applicant asserts that McManis contains no teaching or suggestion of the limitation wherein the loader program is further arranged to verify the digital signature of the at least one administrator-configurable file using the private key and does not disclose an administrator-configurable file." Thus, it is respectfully submitted that claims 5, and 11 are allowable over the prior art of record (Remarks, page 13).

For example, McManis (column 1, lines 10-27) discloses "In the context of a computer program, an "external function" is typically a procedure or function located in a library or other repository of functions that is external to the computer program using the external function. External functions are often, but not always, authored by different people or by different entities than the computer programs using those external functions. Program execution environments that allow external functions to be bound at run time, rather than at link time or compile time, facilitate the maintenance and updating of computer programs because for such programs to be used in such execution environments only the computer programs that are being revised or updated need to be recompiled, while the other modules can be left unchanged. Furthermore, the recompilation process is simplified because the compilation of the revised programs can be performed even if other modules used by the program are not present in the program development system." The Patent Office is requested to identify where in the passages of column 1, lines 10-27, of McManis, there is disclosure of an administrator-configurable file.

Finally, the Patent Office asserted (page 12, lines 7-11, of the Office Action mailed March 07, 2006) "In light of the reasons of record, the examiner finds the arguments of the applicant's representative to be unpersuasive. Furthermore, applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

In response, Applicant notes that the above paragraphs are replete with claimed subject matter that appear not to be either disclosed or fairly suggested by McManis. If the claimed subject matter is not disclosed or fairly suggested by McManis, then the claimed subject matter is new and not obvious. The claimed subject matter is also useful in the context of the claimed invention. Thus, Applicant throughout prosecution has been pointing out how the language of the claims patentably distinguishes them from the references.

The Examiner is respectfully requested to reconsider and remove the rejections of the claims 1, 2, 5, 7, 8, 11, and 13-15 under 35 U.S.C. 103(a) based on McManis, U.S. Patent No. 5,757,914 and to allow all of the pending claims 1, 2, 5, 7, 8, 11, and 13-23 as now presented for examination. An early notification of the allowability of claims 1, 2, 5, 7, 8, 11, and 13-23 is earnestly solicited.

S.N.: 10/050,083
Art Unit: 2137

Respectfully submitted:

Walter J. Malinowski August 24, 2006

Walter J. Malinowski Date

Reg. No.: 43,423

Customer No.: 29683

HARRINGTON & SMITH, LLP

4 Research Drive

Shelton, CT 06484-6212

Telephone:	(203)925-9400, extension 19
Facsimile:	(203)944-0245
email:	wmalinowski@hspatent.com

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. BOX 1450, Alexandria, VA 22313-1450.

8/24/06 Ann Okrentowich

Date Name of Person Making Deposit